International Journal of Innovative
Computing, Information and Control
Volume **15**, Number **5**, October **2019**

ICIC International ©2019 ISSN 1349-4198
pp. 1935–1946

LICODS: A CNN BASED, LIGHTWEIGHT RGB-D SEMANTIC SEGMENTATION FOR OUTDOOR SCENES

Ali Suryaperdana Agoes[1], Zhencheng Hu[2] and Nobutomo Matsunaga[1]

[1]Department of Information Technology on Human and Environmental Science
Kumamoto University
2-39-1, Kurokami, Chuo-ku, Kumamoto 860-8555, Japan
bledeg99@gmail.com; matunaga@cs.kumamoto-u.ac.jp

[2]Wissen Intelligent Sensing Technology
7F, Block D, BenQ Business Building, No. 207, Songhong Road, Shanghai 200000, P. R. China
jameshu2015@hotmail.com

Received March 2019; revised July 2019

Abstract. *One way to visually understand the scenes is through per-pixel semantic segmentation. Recently, this field has undergone heavy development following the recent success of feature learning method based on the Convolutional Neural Network (CNN). Encouraged by this observation, we present Lightweight Color Depth Semantic Segmentation (LICODS), a small numbered parameter model based on the CNN for RGB-D image semantic segmentation. Additional input modality besides the color information to enhance per-pixel class prediction accuracy is employed. On the other side, our model parameter number remains low, although dual branches exist along our model's network. The model performs better compared with the recently published RGB-D semantic segmentation models in terms of accuracy and processing time, despite of its small parameter number.*

Keywords: Pixel-wise classification, Semantic segmentation, Scene understanding

1. **Introduction.** Broadly speaking, the methods to understand the environment fall into two categories. First is image semantic segmentation, a method to reveal pixel information upon the camera's field of view. Second is object detection, a strategy to extract the main and supporting target of interest within the frame. Image semantic segmentation method is divided into: the non-learning and the feature learning based segmentation. The non-learning image semantic segmentation is such as in [1, 2, 3]. The feature learning based segmentation is an effort to mimic the human ability to perceive the environment information which is modeled via the Convolutional Neural Network (CNN), for example, Fully Convolutional Network (FCN) [4], and SegNet [5]. While each scene understanding method offers merit and demerit, our paper will focus on the feature learning based semantic segmentation method. Our main motivation is that CNN based semantic segmentation provides robust pixel classification. Robust pixel's labeling is suitable for assisting autonomous vehicle maneuverability while cruising the road as in [6, 7]. Outdoor environment is prone to weather changes, rapid alteration of ambient light, and partial lighting. Some other practical examples of the image semantic segmentation application are Advanced Driver-Assistance System (ADAS), and surveillance.

Powered by robust feature learning method, the image semantic segmentation problem gains more interest from computer vision researcher community. The prominent landmark achieved by the work of Long et al. [4] is commonly known as FCN. Later on,

DOI: 10.24507/ijicic.15.05.1935

Badrinarayanan et al. in SegNet's architecture [5] introduce the method to transfer the indices of maxpooling layer. The indices are deployed to overcome the memory expenses on the shortcut connection of FCN. Both FCN and SegNet employ the bilateral filter as part of their upsampling layer. This strategy is used to overcome the feature map resolution reduction caused by deployment of pooling layer. The next is the introduction of dilated convolution to solve the pooling layer problem over the image segmentation workflow. Dilated convolution is first used in [8]. Similarly, filter named atrous convolution is deployed in [9]. The excessive deployment of these filters in the deep learning architecture results in the spatial information loss. In particular, the small object's feature missing detail will be impossible to recover after the loss. We highlight that by careful placement and limit upsampling layer number. The relatively small yet important class can maintain its spatial structure during a forward pass.

The photometric information is commonly employed as the raw material input for the CNN based image semantic segmentation. Furthermore, there are efforts to augment input modality taken by researchers to further enhance the model's performance. By using depth for the additional input, this potentially has the contribution for model's pixel-wise prediction improvement. Other than the depth there is another option, which is HHA-feature that was mentioned by Gupta et al. in [10]. HHA-feature denotes an image encoding based on the horizontal disparity, height above the ground, and the angle of the pixels local surface. HHA-feature is basically a pre-computed feature that is geometrically inherited from the photonic data. Whether which feature offers a best result for the model, remains in question. We take the depth option, solely because of less computational demand.

The vast majority of RGB-D image semantic segmentation approaches still lack exceptional outcome toward the RGB mode model. The gap between RGB-D and RGB mode model is what this paper tries to answer. We believe that the depth is a viable option to boost the pixel prediction accuracy. However, the extra mode means the increment of parameter number. In other words, the model will no longer effective for real-life application due to its high computational cost. This paper presents an approach to semantically segmenting RGB-D image in a very efficient way. Our CNN based RGB-D semantic segmentation is coined as Lightweight Color Depth Semantic Segmentation (LICODS).

2. **Related Work.**

The recent success of the feature learning approach for image classification stimulates the semantic segmentation research. FCN [4] is a pioneer among the CNN based image segmentation. This work introduces a method to adopt deep network for pixel level classification. Furthermore, they have conducted experiments with model's modality. They combine the RGB with depth, and RGB with HHA-feature to further generalize their work. The way they adjoint extra input is by means of early and late fusion. The former refers to the early concatenation of color and depth before entering network branch. The latter refers to separately feed dual-input into double identical branches inside a network, and concatenated them at the end.

Hazirbas et al. [11] present FuseNet which had a 16-layer VGG as its model's core. Their model resembles the encoder-decoder type of DeconvNet [12] and SegNet [5] with the further consideration on its encoder to accommodate depth. Unlike the FCN, FuseNet adopts the depth information by element-wise summation. Their approaches are split into sparse and dense fusion, in which, the two are differentiated by fusion layer insertion into the main branch. Our model is inspired by their RGB-D architecture. However, our work is dissimilar to their encoder-decoder style network. Our CNN based semantic segmentation model has not used a sequence of downsampling and is followed by another series of upsampling. Our model simply makes full exploitation of the downsampling to

robustify the extracted features. The process is followed by resolution restoration. It is solely done by one deconvolution layer. Moreover, RGB-D semantic segmentation models we stated previously were not providing the best trade-off between speed and performance.

In image classification trend, deep learning method gains more attention since the CNN reintroduction as in AlexNet [13]. The AlexNet performance in the ImageNet classification challenge performed quite astonishing compared to the other state of the art at that time. Since then, most of the CNN research direction is in the way going deeper architecture aimed for better accuracy. On the contrary, there were groups of researcher trying to reach good performance while keeping in mind the model's layer configuration. The size of the network, specifically parameter size, has a significant role in the computational cost of a CNN based semantic segmentation. It is directly proportional to computing power requirement. The smaller its size, the less power to compute. And not to mention other benefits like an efficient memory transferability. Four different approaches are introduced as a method for compacting network architecture size, which are matrix factorization [14, 15], weight pruning [16], quantization [17], and small architectural design [18]. In the fashion of small model, Iandola et al. [18] propose SqueezeNet, a light deep network that performs comparable to the well-known model, AlexNet. For comparison, the memory parameter size of SqueezeNet model is 4.8 MB while AlexNet is 240 MB.

The way of SqueezeNet cuts down deep learning architecture's parameter mainly inspires us to do similar approach for the RGB-D semantic segmentation problem. SqueezeNet contains several blocks of fire module. That consists of squeeze and expand layers. While the expand layer was used to capture the feature from the image, the squeeze layer is employed to decrease the number of parameters by utilizing 1×1 kernel. The 1×1 kernel is used to limit the number of input channels to the following layer. The way squeeze layer limits the parameter influences us in our model's architecture design. However, we consider putting a 1×1 kernel in the middle of four 3×3 convolution layer series. Other than that, our score layer employs a 1×1 size kernel of convolutional layer with the output describing the class of a desired prediction. It is inserted after the first block of LICODS. This act is used to constrain the parameter by limiting the depth number by the size of class prediction.

3. **Design Strategies.** In this section we convey our strategies to solve the RGB-D image semantic segmentation problem. We start with our intuition to combine both RGB and depth input. We follow the assumption given by Hazirbas et al. [11]. They assume that the fusion of activations of color and depth branches after the activation layer produces stronger signals. They present the idea to fuse color and depth, while putting aside trivial approach by stacking its array for the given model's input. Their assumption can be written in Equation (1).

$$x_k^{(l+1)} \leq \max\left(0, \left\langle u_k^{(l)}, a^{(l)} \right\rangle + c_k^{(l)}\right) + \max\left(0, \left\langle v_k^{(l)}, b^{(l)} \right\rangle + d_k^{(l)}\right) \tag{1}$$

where x is a set of input, a and b are features learned from the color and depth channels, on the k-th feature map in the l layer, while u, v are decomposition of parameters, and c, d are decomposition of biases. Based on this inequality, the combinatorial branches activation yields a substantial amount pulse to trigger the respectful layer while preserving the earlier layer activation. Both color and depth are regarded having the potential to activate the neuron. Namely, the feature extracted during the learning process complements to each other. This approach is reflected in our architecture design. We assign an equal layer depth for each branch. In the practice, area inside the frames lacks the light intensity, i.e., due to illumination problem. The depth information takes place in the feature learning sessions via the element-wise summation. We further assert that the complementary input

modality yields stronger pixel prediction accuracy. In Section 5, we show our experiment result toward various modalities fed into the network.

The decision to fuse color and depth, leads to a problem for keeping the overall parameter's size small. While depth branch deployment is expected to raise its performance. Its utilization has an impact on the computational cost due to the increasing load during training and inference sessions. In order to realize our approach, our model architectural design relies upon minimizing parameter deployment, see Figure 1. The network is mainly composed of the convolutional layer (shown in yellow rectangle) followed by ELU activation layer (shown in green rectangle). The intra branch features maps combination is achieved by the concatenation layer (shown in dark blue rectangle). We also make use of the maxpooling layer (shown in red rectangle) along with convolutional layer. The fusion between color and depth branch is represented by dashed line fed into the element-wise summation layer (shown in orange rectangle). And, lastly the dropout layer (shown in light blue rectangle). Furthermore, the parameters minimization is accomplished through the following strategies.

- **Strategy 1.** The input has not undergone heavy down-sampling with the consideration, keeping of maximum number of layer's depth low.
- **Strategy 2.** Double 3×3 filters followed by 1×1 filter. The main structure inside of each branch consists of input splitting. It split into the two separate branches.

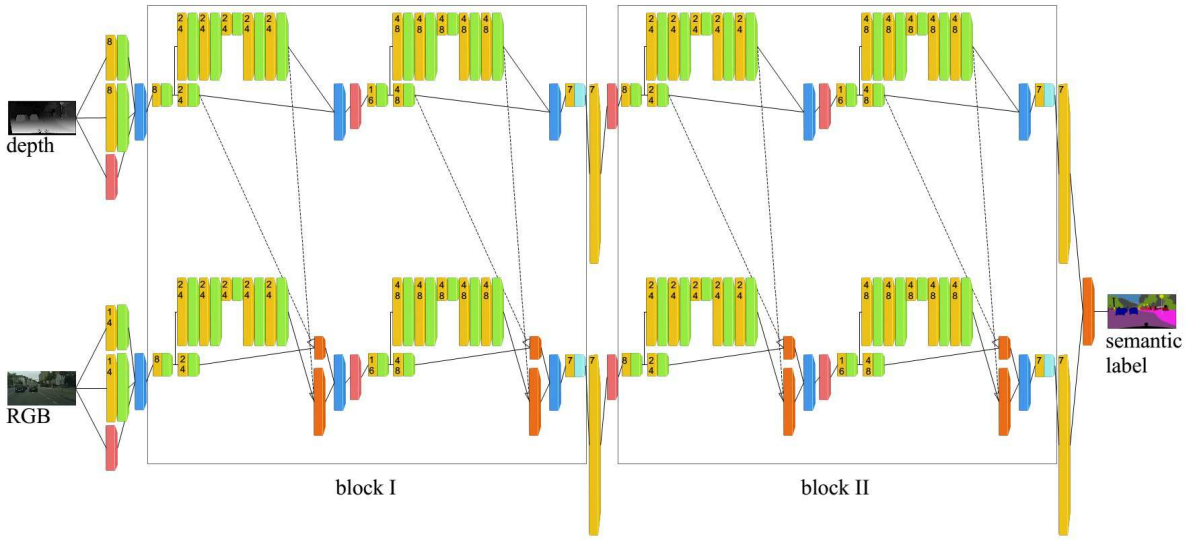

FIGURE 1. (color online) LICODS architectural design consists of two main branches to accommodate the RGB-D input, best viewed on-screen or color printing. The upper side of block diagram is the depth branch. The lower side of the block diagram is the color branch.

Our first strategy is contradictory with the commonly deployed model for semantic segmentation. The concept presented in FCN [4] is to adapt CNN for image segmentation problem. FCN's feature maps are learned to represent an object downscaled from its original size. Before being restored to fully predict each pixel's class. In order to recover the spatial information from the earlier pooling layer, they utilize a skip connection. The SegNet's [5] approach to recover the loss information is by passing the early pooling layer's indices of its respective size. It is then sent to the upsampling layer to combine them. While these seem to be an apparent work around for the spatial information loss problem. The parameter increment could not be avoided. Our approach is to restrict the

downsampling peaked at four times smaller than its input size, see our list in Table 5 for detail of output size after each layer. As the trade-off, model's learning and inference load will increase. This is due to the amount of RGB-D resolution size input's array that need to be processed. We solve the problem by suppressing the depth's of convolution filter size. Our model architectural design deploys a fairly small filter's depth. The layer depth contributes significantly to the size of the parameter. This is because the next convolution process involves the depth size of the previous layer for its input. Our strategy of depth layer assignment is displayed as a number on the boxes in Figure 1, which depict the depth of each convolution layer.

The next strategy is the utilization of double 3×3 followed by a 1×1 and double 3×3 kernel. Although 5×5 kernel gives bigger receptive size, it is undesirable due to its computational cost. The 3×3 kernel operates on the equal size with less processing power. Further, we insert 1×1 size filter for separation between the double 3×3 kernel. This will allow the subsequent 3×3 filter to see the output map from the previous double 3×3 convolution layer. Next, we concatenate the feature maps with the early branch division. We find that this combination gives stronger feature extraction on the small object. On the network perspective, RGB and depth branch are separated and then merged by element-wise summation, shown by the dashed line in Figure 1. We discover that, our approach works well to activate neurons in the state of weaker object's RGB definition input.

Figure 1 shows two separate blocks, which are block I and block II. The architecture is composed of two major blocks, we refer block I as the LICODS Single Block (SB), and block II as the LICODS Double Block (DB). Each block consists of similar layer and number of filters. Both blocks are connected by a score and an upsampling layer. The score layer is a convolution stage with 1×1 kernel size and output of seven layer depth. We assign seven layer depth because our model predicts seven different objects for each pixel. The score layer allows further reduction on parameter dimensionality. This is accomplished by similar way of limiting the depth of input size for the next layer in the model architecture. Furthermore, we restore the feature map to its original resolution size using the deconvolutional layer. Here, deconvolutional layer uses a constant value bilinear filter. We find this works better toward the dynamic value filter. With an extra LICODS SB block after the feature maps are restored to its original resolution. We observe better generalization due to the increment of model's depth, i.e., it makes the receptive field grow larger.

LICODS utilizes the Exponential Linear Unit (ELU) as the activation layer. Deployment of ELU [19] in replace of the Rectified Linear Unit (ReLU) and the BatchNorm layer combination. Consistent with its result reported in the paper, we gain similar model behavior toward neuron activation. The ELU layer, forces the activation move centralized to zero mean. This implies to the faster learning rate and higher prediction accuracy.

4. **Experiment.** In this section, we deliver our experimental result. We conduct our experiment using the publicly available Cityscapes dataset [20]. The dataset contains pixel-wise annotated synchronized RGB-D images of outdoor scenes. It is divided into finely and coarsely annotated segmentation. In this experiment, we use the fine annotation RGB-D images, which consists of 2975 training images, 500 validation images, and 1525 testing images. We use the standard train-val-test splitting for our experiment. The database comes with 34 classes and 8 categories. We adjust the number of classes into seven to fit with our experiment. The class's labels are road, sidewalk, building, vegetation, sky, person, and car. We set other than the intended class to void during the training.

Our experiments are performed in the end-to-end manner, where all the network architectures undergo similar treatment to obtain the comparable result. We train our model from scratch with the optimum adjustment of hyper-parameter. We use the GTX 1060 platform, which has the memory limitation of 6 Gigabytes. By fact that each of architecture requires the RGB and depth to fed, thus we downsized the original cityscape resolution, from 2048×1024 to 256×128. The training and testing are carried out in the Caffe framework [21]. We train the models using Stochastic Gradient Descent (SGD) solver using batch size of 1. The RGB-D images randomly enter the architecture, with the consistent random generation number for each model. We employ different learning rate initialization besides the fix learning rate applied to all sessions. In particular, compare parameter learning rate effectiveness. We deploy the same 20 epoch training across all the models we tested. We utilize a high momentum of 0.99 and weight decay of 0.0005. The RGB and depth training set are then subtracted with its respective mean value to push the distribution centered to zero. Furthermore, the depth value is calculated from disparity provided in the cityscapes dataset. We further normalize the depth data to the value 0-255.

5. **Results.** We evaluate the trained models over 500 validation images. We use pixel accuracy, mean accuracy and mean Intersection over Union (mean IU) to quantitatively measure the model's performance similar to the metrics reported in [4].

- pixel accuracy: $\sum_i n_{ii} / \sum_i t_i$
- mean accuracy: $(1/n_{cl}) \sum_i n_{ii}/t_i$
- mean IU: $(1/n_{cl}) \sum_i n_{ii} \Big/ \left(t_i + \sum_j n_{ji} - n_{ii}\right)$

We test our LICODS behaviour on the three schemes to show the different input mode impact. First scheme, we use RGB-D input for the LICODS DB. Second scheme, we eliminate the depth branches. We use solely the RGB images as the input to train the model. Our purpose is to reveal the depth's branch contribution to the element-wise summation layer. The last scheme, we use the full branch LICODS DB RGB-D. Here, we substitute the calculated depth with the grayscale images which is converted from the original RGB images. Additionally, we also test the LICODS SB to show the trade-off between the parameters and performance to LICODS DB.

We present the LICODS DB performance on three different schemes in Table 1. Our best performance on mean IU is achieved by the RGB-D input combination. The extra depth branch boosts the model's pixel prediction by 12% over the model utilized solely on RGB. Also, it is 10% better compared to the model using grayscale as a depth replacement. Further, we cater our result of testing LICODS single and double block deployment. We gain 2.2% improvement by doubling the LICODS's architectural network block. As the trade-off of its performance, the computational cost increased by 2.36 ms against the

TABLE 1. Our LICODS performance on different input modality. The additional depth branch significantly improves the performance by 10% over the gray-scale replacement and 12% over the model deploying only RGB input.

Model/Scheme	Pixel Acc.	Mean Acc.	Mean IU
LICODS DB RGB-D	**93.0**	**87.7**	**78.7**
LICODS SB RGB-D	91.9	86.4	76.5
LICODS w/o depth	88.9	76.6	66.9
LICODS gray subs	89.4	78.6	68.8

TABLE 2. Comparison of average inference time (in ms), and the number of parameters. We compare the time taken for each model to forward pass the 256×128 resolution of RGB-D input. We conduct the experiment using the GTX 1060 platform.

Model Name	Ave. Inf. Time	Parameters
LICODS SB	**4.64 ms**	**293.01k**
LICODS DB	7 ms	382.31k
FCN 32s early	40.49 ms	134.49M
FCN 32s late	79.16 ms	268.78M
FuseNet SF1	29.63 ms	23.99M
FuseNet SF5	38.1 ms	38.67M

TABLE 3. Performance comparison of LICODS DB, FCN 32s, and FuseNet. LICODS DB gives 6.8% and 9.1% improvement over the FuseNet SF5 and FCN 32s early fusion, respectively.

Model Name	Pixel Acc.	Mean Acc.	Mean IU
LICODS SB	91.9	86.4	76.5
LICODS DB	**93.0**	**87.7**	**78.7**
FCN 32s early	89.5	81.1	69.6
FCN 32s late	88.8	80.1	68.0
FuseNet SF1	91.1	79.7	71.4
FuseNet SF5	90.1	81.7	71.9

LICODS SB. This result is by far faster than the FuseNet and FCN. The way we calculate the inference time is by averaging single forward pass on a group of test images that imply to all the models. Table 2 shows the complete comparison of average inference time and the number of parameters from LICODS, FCN [4], and FuseNet [11].

In order to further analyze the performance of LICODS's pixel-wise prediction. We compare LICODS with other RGB-D semantic segmentation methods. Table 3 demonstrates the result of LICODS SB and DB, and its comparison to FCN 32s early fusion, FCN 32s late fusion, FuseNet SF1, and FuseNet SF5. Despite the FuseNet merges the RGB and depth in the similar fashion, LICODS DB gives 6.8% better performance in perspective of the mean IU, compared to the FuseNet SF5. Arguably, our approach to constraining feature resolution downsampling shows its impact. However, the LICODS performance is not further improved as for introducing another block. Next, in Figure 2 we show the effect of iteration number to the convergence rate. We constrict our training to 20 epochs, and apply this to all models that we have used for comparison. We find that the LICODS learning rate is faster, although we use the same weight initialization across all the models we tested.

We analyze LICODS for class-wise prediction accuracies. In Table 4, we report per-pixel class prediction for each model. As the result of minimum downsizing during the feature learning steps. The output image resolution requires less restoration. Therefore, for the smaller object like person, LICODS DB achieves 5% better accuracy across the tested models. Similarly, for the car class LICODS DB got 4.1% improvement. For road object, that occupied a bigger portion in the image area, LICODS DB achieves a small improvement compared to its close contender the FuseNet SF5. Figure 3 demonstrates the image's segmentation result for visual assessment to compare LICODS with FuseNet and FCN 32s. Overall, the LICOD DB's performance outperforms other RGB-D semantic

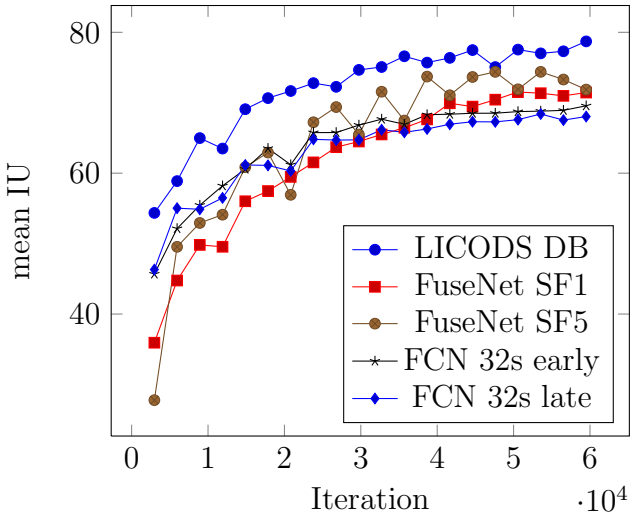

FIGURE 2. Training from the scratch for each model in 20 epochs. We show the parameter learning rate effectiveness of LICODS compared to FCN and FuseNet.

TABLE 4. Class-wise segmentation accuracies of seven classes. We compare LICODS to the model trained with RGB-D input.

	Road	Sidewalk	Build	Veg	Sky	Person	Car
LICODS SB	92.9	61.3	84.5	85.2	85.1	48.1	78.7
LICODS DB	**94.4**	**67.4**	**86.1**	**85.3**	**87**	**49.6**	**81.2**
FCN 32s early	92.5	58.8	79.4	76.7	71.5	34	73.9
FCN 32s late	91.7	55.4	79.1	76.3	71.1	31.8	70.9
FuseNet SF1	93.6	61	81.9	80.5	82	24	77.1
FuseNet SF5	91.5	48.1	82.2	79.1	85	44.2	73

segmentation models. However, the performance outcome is still not perfect. One of the examples (in Figure 3), LICODS DB segmentation's result is shown in the fourth row. Although, the ground truth depicts the walker as a void, LICODS DB determines as a person. Similarly, the bikes on the sidewalk in LICODS DB segmentation's result is determined as a person. Both of the examples, do not yield a dangerous situation as the void miss classified as the road.

Our deep learning architecture input modality is not confined to only photometric information. Despite been faced with those challenges, our model is able to tackle the hindrances. See Figure 4, the chart shows our results. The figure shows number of parameters, prediction accuracy and processing speed. It compared LICODS to the state of the art CNN based RGB-D semantic segmentation method. The small number parameter of our architecture does not limit its performance. Rather, LICODS outperforms other CNN based RGB-D semantic segmentation in terms of accuracy and processing time.

Figure 5 to illustrate our semantic segmentation result visually projected into 3D world coordinate. The picture shows perspective toward its surrounding via front ego vehicle view. The "road" label represents the available free space. The segmentation result is not only revealing precise approximation of the space where vehicle is possible to cruise. Other than that, the exact type of present obstacle ahead is inferred.

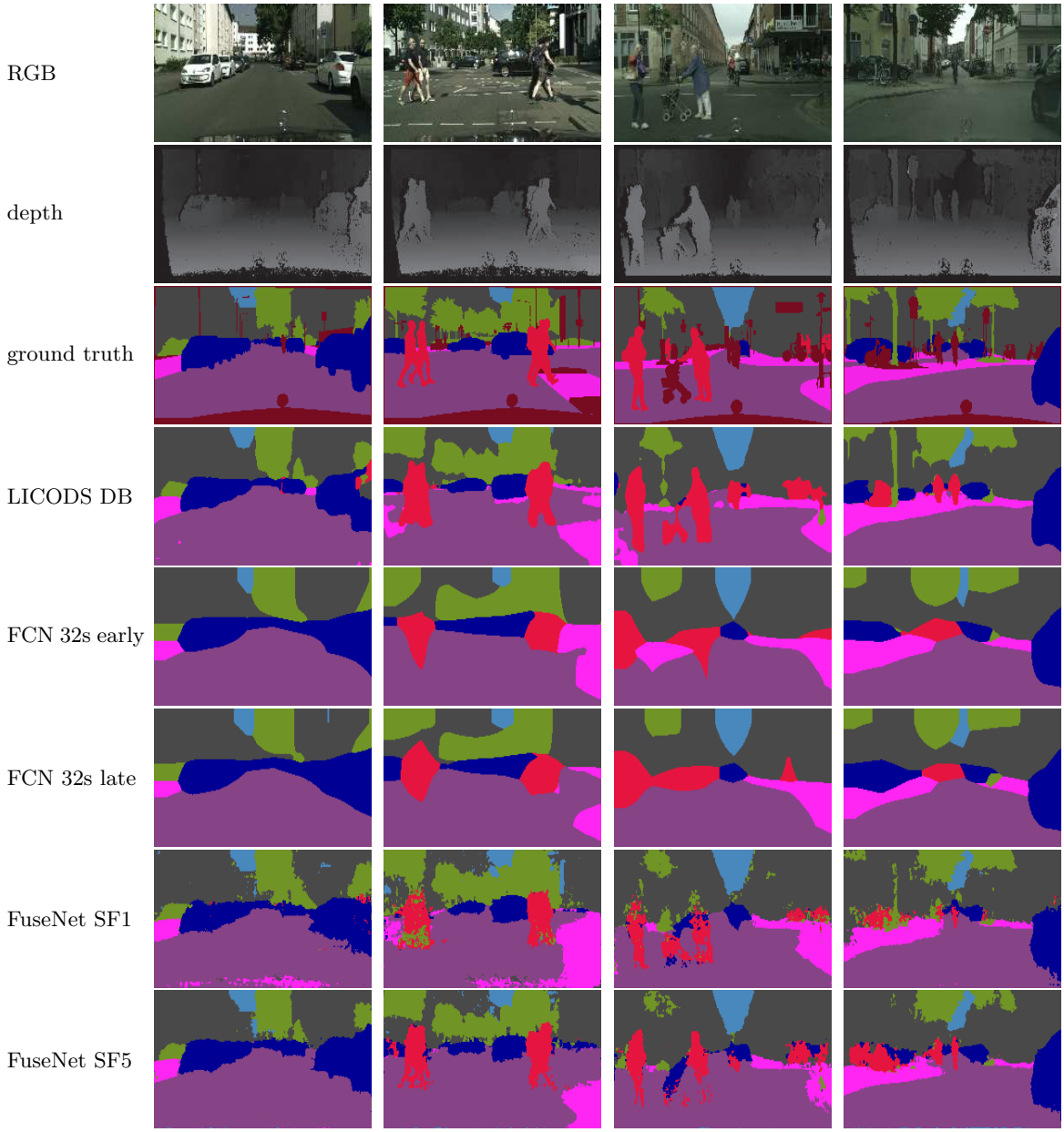

FIGURE 3. (color online) Qualitative segmentation results for different architectures. Color legend: road, sidewalk, building, vegetation, sky, person, car.

6. **LICODS Architectural Exploration.** To this end, we have explored our architectural design strategies followed by its result compared to the other RGB-D semantic segmentation based on CNN. In this section, we analyze the block consisting of layers that construct LICODS. Table 5 shows LICODS layer's name, output size and the parameter number. Our model starts with three filters that have a direct contact with each of the input, either RGB or depth image. We depict this block as the detection block. It consists of a maxpooling and two convolutional layers. The convolutional layers and maxpooling layer are using two pixels stride. Therefore, the size of the input reduced to half of its original size after passing the initial layer of detection block. The combination of two different size convolutional layers and a maxpooling layer is meant to extract the meaningful representation of the object. Subsequently signals are concatenated and fed

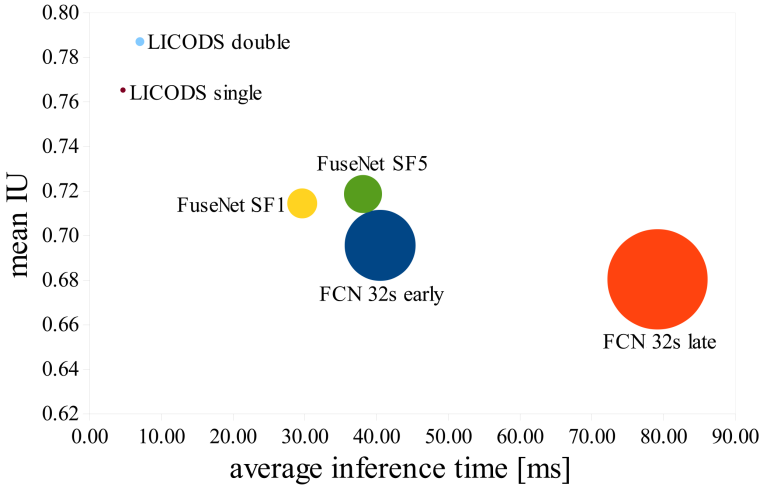

FIGURE 4. LICODS comparison in terms of mean IU vs. average inference time and number of parameters. The size of blobs is an illustration for the number of parameters.

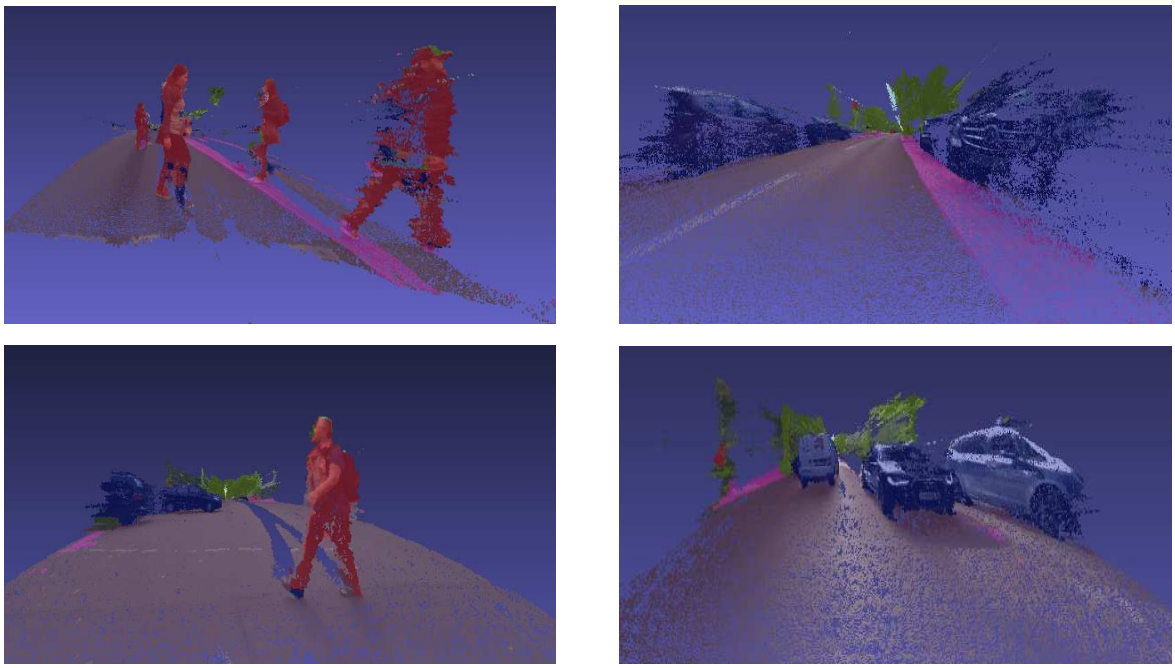

FIGURE 5. (color online) 3D projection of LICODS's semantically segmented object. The color represent each class used in the experiment. Color legend: road, sidewalk, building, vegetation, sky, person, car.

into the following layer. The concatenation is intended for supplying the model with the appropriate features to activate the neurons.

The LICODS's core block started with 1×1 convolutional layer. Then, it fed two branches with feature maps. First branch consists of double 3×3 convolutional layer, 1×1 convolutional layer and double 3×3 kernel convolutional layer. Second branch consists of only 1×1 kernel. The subsequent 1×1 kernel convolutional layer will collect all inputs and collapses into one before entering the next double 3×3 kernel. The overall LICODS undergoes two times downsampling, one parameter reduction and one upsampling on each

TABLE 5. LICODS DB architectural description of layer name, output size and parameters

Layer name/type	Output size	Parameter	Layer name/type	Output size	Parameter
RGB input	256×128×3	–	fire2/squeeze1×1 d	128×64×8	144
block1/rgb pool	128×64×3	–	fire2/exp1×1 d	128×64×24	216
block1/conv rgb 2×2	128×64×14	182	fire2/exp3×3 d 1	128×64×24	1,752
block1/conv rgb 3×3	128×64×14	392	fire2/exp3×3 d 2	128×64×24	5,208
block1/concat rgb 1	128×64×31	–	fire2/m.s.exp3×3 d 2	128×64×24	600
Depth input	256×128×1	–	fire2/exp3×3 d 3	128×64×24	5,208
block1/d pool	128×64×1	–	fire2/exp3×3 d	128×64×24	5,208
block1/conv d 2×2	128×64×8	40	fire2/concat rgbd	128×64×48	–
block1/conv d 3×3	128×64×8	80	pool rgbd 2	64×32×48	–
block1/concat d 1	128×64×17	–	total fire 3	–	146.53k
fire2/squeeze1×1 rgb	128×64×8	256	total fire 3 score	–	1,358
fire2/exp1×1 rgb	128×64×24	216	total upscore 3	–	6,272
fire2/exp3×3 rgb 1	128×64×24	1,752	total fire 4	–	36.51k
fire2/exp3×3 rgb 2	128×64×24	5,208	total fire 5	–	146.53k
fire2/mse3×3 rgb 2	128×64×24	600	total fire 5 score	–	1,358
fire2/exp3×3 rgb 3	128×64×24	5,208	total upscore 5	–	6,272
fire2/exp3×3 rgb	128×64×24	5,208	**total parameter**	–	**382.31k**

block. The total parameter for LICODS DB is 382.31k by assuming that only convolution and deconvolution layers have contributed to parameter calculation.

7. **Conclusion.** In this paper, we have presented LICODS, a CNN based RGB-D semantic segmentation architecture. The proposed model exploits the CNN capability to do the pixel-wise class prediction in the small parameter fashion. By conducting the experiments followed by the careful evaluation, we may conclude that our solution outperforms the FCN 32s and FuseNet for RGB-D semantic segmentation regardless its small parameter number. The direction for future research is clear ahead. A supervised CNN based image semantic segmentation is a labor intensive work for ground truth preparation. It is desired to design an efficient RGB-D semantic segmentation model with semi-supervised or even un-supervised training.

Acknowledgment. I would like to express my gratitude to Zhencheng Hu, Ph.D., the CEO of Wissen Intelligence Sensing, for giving me the opportunity to do an internship within the organization. For me it was a unique and valuable experience to get involved in the industrial sector of Intelligent Transportation Systems (ITS). I would also like to thank all the people that worked in the office of Wissen Auto Sensing in Fukuoka especially Mr. Hirano and Mr. Ly, also, member of R&D team of Wissen Intelligence Sensing Shanghai. With their patience and openness, they created an enjoyable working environment. Lastly, I would like to express my gratitude to Prof. Nobutomo Matsunaga, for allowing me to join Wissen as an intern.

REFERENCES

[1] J. Yang, T. Liu, R. Chen, G. Zhang, H. Peng and J. Wang, A kernel-based clustering algorithm under the framework of membrane computing for image segmentation, *ICIC Express Letters, Part B: Applications*, vol.10, no.4, pp.311-317, 2019.

[2] M. Delić, L. Nedović and E. Pap, Extended power-based aggregation of distance functions and application in image segmentation, *Information Sciences*, 2019.

[3] Y. Yang and W. Jia, Improved level set model based on bias information with application to color image segmentation and correction, *Signal, Image and Video Processing*, pp.1-9, 2019.

[4] J. Long, E. Shelhamer and T. Darrell, Fully convolutional networks for semantic segmentation, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.3431-3440, 2015.
[5] V. Badrinarayanan, A. Kendall and R. Cipolla, SegNet: A deep convolutional encoder-decoder architecture for image segmentation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.39, no.12, pp.2481-2495, 2017.
[6] M. Treml, J. Arjona-Medina, T. Unterthiner, R. Durgesh, F. Friedmann, P. Schuberth, A. Mayr, M. Heusel, M. Hofmarcher, M. Widrich et al., Speeding up semantic segmentation for autonomous driving, *The 29th Conference on Neural Information Processing Systems (NIPS)*, Barcelona, Spain, 2016.
[7] G. Ros, L. Sellart, J. Materzynska, D. Vazquez and A. M. Lopez, The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.3234-3243, 2016.
[8] F. Yu and V. Koltun, Multi-scale context aggregation by dilated convolutions, *arXiv Preprint*, arXiv:1511.07122, 2015.
[9] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy and A. L. Yuille, DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.40, no.4, pp.834-848, 2018.
[10] S. Gupta, R. Girshick, P. Arbeláez and J. Malik, Learning rich features from RGB-D images for object detection and segmentation, *European Conference on Computer Vision*, pp.345-360, 2014.
[11] C. Hazirbas, L. Ma, C. Domokos and D. Cremers, FuseNet: Incorporating depth into semantic segmentation via fusion-based CNN architecture, *Asian Conference on Computer Vision*, pp.213-228, 2016.
[12] H. Noh, S. Hong and B. Han, Learning deconvolution network for semantic segmentation, *arXiv Preprint*, arXiv:1505.04366, 2015.
[13] A. Krizhevsky, I. Sutskever and G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Advances in Neural Information Processing Systems*, pp.1097-1105, 2012.
[14] Y. Gong, L. Liu, M. Yang and L. Bourdev, Compressing deep convolutional networks using vector quantization, *arXiv Preprint*, arXiv:1412.6115, 2014.
[15] J. Jin, A. Dundar and E. Culurciello, Flattened convolutional neural networks for feedforward acceleration, *arXiv Preprint*, arXiv:1412.5474, 2014.
[16] S. Han, J. Pool, J. Tran and W. Dally, Learning both weights and connections for efficient neural network, *Advances in Neural Information Processing Systems*, pp.1135-1143, 2015.
[17] S. Gupta, A. Agrawal, K. Gopalakrishnan and P. Narayanan, Deep learning with limited numerical precision, *International Conference on Machine Learning*, pp.1737-1746, 2015.
[18] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally and K. Keutzer, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and $<$ 0.5MB model size, *arXiv Preprint*, arXiv:1602.07360, 2016.
[19] D. A. Clevert, T. Unterthiner and S. Hochreiter, Fast and accurate deep network learning by exponential linear units (ELUs), *arXiv Preprint*, arXiv:1511.07289, 2015.
[20] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth and B. Schiele, The Cityscapes dataset for semantic urban scene understanding, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
[21] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama and T. Darrell, Caffe: Convolutional architecture for fast feature embedding, *arXiv Preprint*, arXiv:1408.5093, 2014.